

## Clock Recovery PLL

### BACKGROUND OF THE INVENTION

#### Field of the Invention

- This invention relates to packet switched networks, and more particularly to a method of
- 5 clock recovery in cell-relay networks, particularly ATM (Asynchronous Transfer Mode) networks offering constant bit rate services.

#### Brief Description of the Prior Art

- AAL1 is the ATM Adaptation Layer for Constant Bit Rate (CBR) services. CBR services carry time sensitive data, such as voice and video. The original clocked (isochronous)
- 10 data is segmented into ATM cells at the transmitting end, carried over the ATM network, and reassembled into a continuous clocked stream at the receiving end. The receiving end needs to provide a clock for this output data stream, or recover the transmitting end clock.

Various standardized and un-standardized clocking schemes are current used in ATM:

- 15 1. Line Rate Clocking: A clock is provided at the receiving end by an incoming data link with the same nominal clock rate as the transmitting end. This incoming clock will often need to be de-jittered before being used, because requirements on outgoing telecom clocks are usually more stringent than requirements on incoming telecom clocks.
- 20 2. Network Synchronous Clocking: A telecom clock is recovered from an 8 kHz reference clock. This is preferably a clock which is PRS-traceable (traceable to a Primary Reference Source). Standards for SDT (Structured Data Transfer) AAL1 require that a clock be recovered from the ATM Network clock, which is a multiple of 8 kHz. The 8 kHz must be frequency-multiplied up to the desired rate.
- 25 3. Adaptive Clocking: The clock from is recovered from information about the fill level of the receive buffer containing the received segmented data. This buffer serves to even-out the variation of the incoming data versus the receiver-end clock which is reading out the data. This method is referred to in the AALI

standards as "Adaptive" but the exact method used is not subject to standardization.

4. SRTS Clocking: The clock is recovered from time stamps carried by the ATM cells. The method standardized for AALI is referred to as Synchronous Residual Time Stamps (SRTS) and described in US patent no. 5,260,978.

In the prior art, external PLL(s) as employed and an algorithm resident in CPU was used adaptive clocking and SRTS.

An object of the present invention is to provide a way of efficiently dealing with all four of these required clocking schemes in one circuit implementation.

#### 10 SUMMARY OF THE INVENTION

According to the present invention there is provided a multimode clock recovery circuit for providing constant bit rate services in a cell relay network, comprising an embedded digital phase locked loop (PLL) including an input circuit capable of generating a phase signal from at least two types of input signal, said phase signal controlling the output of said phase locked loop to generate clock signals for said constant bit rate services.

ATM Forum af-vtoa-0078.000 and ITU-T 1.363.1 standards require clock recovery using either the Synchronous Residual Time Stamp or adaptive technique. The invention can recover a service clock with the appropriate quality from an incoming clock, received SRTS data or FIFO pointer data. The same PLL is used for all three kinds of inputs thereby providing a gate efficient implementation.

The present invention is primarily concerned with telecom clocks at T1, E1 and J2 rates, as well as TDM bus clocks at 2.048 MHz, 4.096 MHz and 8.192 MHz. The invention is also applicable to DS3 and E3 clock rates.

The clock recovery circuit in accordance with the invention efficiently supports all four clock schemes. The SRTS scheme recovers phase of original clock, not just the frequency

The Adaptive mode works with Dynamic Bandwidth Circuit Emulation Services (DBCES). With DBCES, the number of DS0 64kbps channels carried by an ATM VC (virtual circuit) can vary (e.g. as voice calls are connected and disconnected). This will vary the inter-arrival times of the cells, as the cells contain either 46 or 47 bytes, but the

number of DSO channels carried by the 46 or 47 bytes vary. This invention can correctly recover timing from the Receive Buffer even as the cell inter-arrival times change.

The invention can also offer an accurate holdover mode in the STRS mode, which is controllable by information derived from received ATM cells. For example, an SRTS time stamp may be detected to be invalid because it is protected with error-detection fields. When the receiver/reassembler flags a SRTS nibble as invalid, the PLL goes into a temporary holdover mode to maintain the clock signal.

The PLL also goes into accurate holdover when the receiver/reassembler buffer underruns due to a lost connection. (When the Receive buffer runs out of TDM data in Adaptive Clocking mode, the PLL is sent a "slow-down" signal, when the ATM VC cell stream stops for a user-programmable length of time, the PLL is automatically put in holdover mode)

The PLL goes into accurate holdover when SRTS cell stream stops (when Receive buffer runs out of new SRTS values, the PLL will automatically go to holdover)

In the Line Rate mode, the PLL to go into accurate holdover mode when the input data link is determined to be in Loss Of Synchronization state.

When the Dynamic Bandwidth Circuit Emulation Services is used with Adaptive mode, and all DSO channels are disconnected (the all-idle state), only a few cells are transmitted each second to keep the ATM link alive, but no TDM (Time Division Multiplexed) bytes are transferred. The clock recovery circuit of the invention will automatically go into accurate holdover mode when the all-idle state is detected as defined by the DBCES channel activity bit mask.

The invention can also provide an accurate freerun clock (0.1 ppm with an accurate MCLK master clock), reduced output jitter, large input jitter acceptance (configurable in gates). It also performs SRTS for UDT (standardized) and also SRTS for SDT (non-standard).

The ability to do Line Rate and SRTS Clocking in one PLL means that at the transmitting end the present invention can be used to de-jitter the incoming clock before it is used to generate SRTS time stamps, and at the receiver another instance of the present invention can be used to generate the outgoing clock from the SRTS. This is important, because the

sampling effects of the time stamp process will alias the jitter frequencies downward, where they will be harder to filter for the receiver PLL (creating wander).

#### BRIEF DESCRIPTION OF THE DRAWINGS

The invention will now be described in more detail, by way of example only, with reference to the accompanying drawings, in which;-

Figure 1 is a block diagram showing adaptive line rate clocking;

Figure 2 is a block diagram showing adaptive clock recovery;

Figure 3 is a block diagram showing SRTS clock recovery;

Figure 4 is a block diagram showing normal transmit SRTS;

10 Figure 5 is a block diagram showing transmit SRTS de-jittering;

Figure 6 is a block diagram of the digital PLL;

Figure 7 is a block diagram of the SRTS generator;

Figure 8 is a block diagram of the phase detector for SRTS inputs;

Figure 9 is a block diagram of the phase detector for clock inputs;

15 Figure 10 shows a combined phase detector for both kinds of inputs;

Figure 11 is a block diagram of the loop filter;

Figure 12 is a block diagram of the digital controlled oscillator; and

Figure 13 is a block diagram of the jitter reduction circuit;

#### DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

20 In Figure 1, the digital phase locked loop (PLL) 1 receiving a line rate input clocks received data out from receive buffer 2. The incoming ATM cell stream is reassembled in reassembly engine 3.

In the adaptive scheme shown in Figure 2, buffer fill-level information is returned from the receive buffer 2 to the input of the PLL 1.

25 In the SRTS scheme shown in Figure 3, the receive SRTS circuit 4 recovers timing information from the incoming time stamps and provides an input to PLL 1.

In Figure 4, the line rate input is fed to transmit SRTS circuit 6, which generates residual time stamps to be included in the cell stream by segmentation engine 5.

By including the clock recovery PLL, as shown in Figure 5, the circuit can be used to de-jitter an incoming clock before it is used to generate SRTS time stamps, and at the receiver another instance of the circuit can be used to generate the outgoing clock from the SRTS. This is important because the sampling effects of the time stamp process alias the jitter frequencies downward, where they are harder to filter for the receiver PLL, which creates wander.

As noted above, the PLL 1 operates in the following modes:

- 10 • Synchronous mode: The PLL synchronizes to the incoming clock.
- Adaptive mode: Synchronize to the incoming cell stream by monitoring the write and read pointers of the Reassembly Circular Buffer. The PLL receives a phaseword representing:  $\text{phaseword} = \text{write\_pointer} - \text{read\_pointer} - \text{average\_fill}$ .
- 15 • SRTS mode: Synchronize to the incoming Synchronous Residual Time Stamps (SRTS). The PLL output clock is fed into a local SRTS generator to generate a local SRTS that is compared to the received SRTS.
- Network mode. This mode is similar to Synchronous mode. An 8 kHz clock, generated from the 19.44 MHz or 155.52 MHz network clock, is the input for the PLL. The output can be a line rate clock or an SRTS.
- 20 Within the modes listed above, the PLL 1 has another set of basic modes:
- Normal mode: The PLL synchronizes using the appropriate input signal (i.e., clock, SRTS, FIFO phaseword or 8 kHz network clock).
- Holdover mode: The PLL no longer uses the input signal, but holds to its current output frequency. When LOS becomes active, the PLL is switched to holdover mode. The user can also switch the PLL into holdover mode.
- 25 • Freerun mode: No input is used. The master clock accuracy determines the output clock accuracy.

The digital PLL 1 must provide a clock synchronous to some input signal. Because there are several types of incoming signals to synchronize to (such as clocks, SRTS data and arriving ATM cells), the PLL needs to be capable of accepting various types of input. The basic PLL architecture, however, is the same for all situations. As shown in Figure 6, it consists of a phase detector 10, a loop filter 11, a digital controlled oscillator 12, an intrinsic jitter reduction circuit 12 and a divider 13.

Parameters such as locking range, center frequency, transfer, accuracy, etc. can be configured/programmed easily in the circuit. In addition, features like freerun and holdover mode can be built in. When the system clock is high enough, no analog circuitry is needed to meet the output jitter requirements.

A divider on the 19.44 MHz or 155.52 MHz network reference (not shown) provides the local residual time stamp, as defined in US patent no. 5,260,978 noted above.

The PLL must generate frequencies with an accuracy as stated in Table 12. The accuracy defines the freerun accuracy and the locking range of the PLL. The accuracy of melk must be incorporated in these calculations. A freerun accuracy of approximately 0.2 ppm and a locking range of approximately 160 ppm is suggested, thereby allowing 100 ppm melk accuracy. For DS1, the locking range is approximately 245 ppm, thereby supporting older systems with a 130 ppm frequency accuracy.

TABLE 12. Center Frequency and Required Accuracy

Clock	Frequency	Accuracy	1 UI
DS1	1.544 MHz	+/- 32 ppm	648 ns
E1	2.048 MHz	+/- 50 ppm	488 ns
C4M	4.096 MHz		244 ns
J2	6.312 MHz	+/- 30 ppm	158 ns
C8M	8.192 MHz		122 ns

The af-vtoa-0078.000 standard with respect to jitter and wander points to several ANSI and ITU-T standards, as summarized in the Table below. The jitter requirements must be met in all clocking modes. The wander requirements may not be met in adaptive mode.

### Relevant Specifications

	Clock	Electrical	Jitter	Wander
	DS1	ITU-T G.703	ANSI DS1.102 ANSI DS1.403 ITU-T G.824	ANSI DS1.403 ITU-T G.824
5	E1	ITU-T G.703	ITU-T G.823	ITU-T G.823
	J2	ITU-T G.703	ANSI DS1.102 JT-G.703a	JT-G.703a,

10 Jitter transfer functions are not specified in the above standards. However, some ITU-T specs. regarding digital multiplex equipment, such as G.743, specify jitter transfer functions. For the MT9042, the requirement is that the jitter transfer corner frequency be between 1.2 and 2.0 Hz (ETSI spec.), with a 20 dB/dec roll off. This range is used for this PLL for all input frequencies and meets the requirements of G.743. The jitter transfer is less than +0.5 dB.

15 The standards from the above Table specify wander/jitter input tolerance and maximum output jitter. For the input wander/jitter tolerance, the low frequency parts are the largest and are therefore the most important for the PLL requirements (see the Table below). With a corner frequency larger than 1.2 Hz, it is assumed that the PLL follows wander below 0.1 Hz. The range of the phase detector is therefore based upon the wander  
20 tolerance at 0.1 Hz and is 32 UI peak-to-peak (+/- 16 UI).

### Minimum Input Wander and Jitter Tolerance

	Clock	A0[Ui <sub>pp</sub> ]	A1[Ui <sub>pp</sub> ]	F0[Hz]F1[Hz] @0.1Hz[Ui <sub>pp</sub> ]
	DS1	28	5	1.2E-5 10 6.2
	E1	37	18	1.2E-5 20 23
25	J2	114	5	1.2E-5 10 15

The maximum output jitter on the output of the PLL as specified in the appropriate standards is listed in the Table below.

### Maximum Allowed Output Jitter

Clock	A[UI <sub>pp</sub> ]	A[ns]
DS1	0.07	45
E1	0.2	98
J2	0.1	16

ANSI DS1.403 and DS1.101 require that the output phase change no more than 81 ns per 1.326 ms. The maximum allowable output phase shift with respect to the ideal reference source is 1  $\mu$ s. Therefore, the slew rate of the DCO input may be no more than 61 ppm of the center frequency setting. The "no more than 81 ns / 1.326 ms" requirement is met for all modes.

Because the period of the E1/DS1/J2/C4/C8 signals is less than 1  $\mu$ s, that requirement is met automatically when the PLL is in synchronous mode. When switching between modes, the 1  $\mu$ s MTIE is not met.

The SRTS generator 14, shown in Figure 7, is the same as the standard solution indicated in ITU standard 1.363.1 and described US patent 5,260,978, which is simple, small and efficient. In master mode, the SRTS generator is needed to generate an RTS (residual time stamp) that is transmitted. In slave mode, the SRTS generator is placed in the feedback loop of the PLL to synchronize on the incoming RTS. The SRTS generator consists of a divider 15, a four bit counter 17 and a register 16. The SRTS is in effect the remainder resulting from dividing the network clock  $f_{nx}$  into the service clock  $f_s$  divided by a suitable number, 3000 or 3008.

The maximum jitter allowed on  $f_s$  and  $f_{nx}$  is such that  $(f_{sx}/f_{nx})\text{MOD}16$  remains constant under all circumstances. Otherwise, the basic principle of the SRTS method is violated.  $f_s$  comes from the PLL and is thereby clean enough. For  $f_{nx}$ , the user is responsible for the quality of the provided network clock. If it's not good enough, it must first be filtered with an external PLL.

The phase detector 10 must extract a phase difference value from its two input signals. There are two cases to consider. In one case the inputs are RTS values as inputs. In the other case the inputs are clock inputs. Both cases require different phase detectors, although certain parts can be shared.



In the case of RTS as input values, the output phaseword is the difference of the input values. However, because both input values come from counters, a modulo function is incorporated in these values. That gives false behaviour when one of the input values wrap around because instead of continually with increasing phase difference, the phase output drops back to its minimum value before increasing again, giving a sawtooth effect. This effect can be cancelled by using the wrap-around event to control an up/down counter, which in effect keeps track of the number of wrap-arounds. This up/ down counter does not wrap around, since otherwise it would re-introduce the same problem. The value of the up/down counter gets a weight of 16 and is added to the SRTS difference. Thus when the first four-bit counter fills up, the up/down counter increments by 1 and adds a weight of sixteen to the four-bit counter output so that the next value from the four -bit counter is treated as seventeen, not one. In this way, the sawtooth effect can be avoided.

The block diagram of the phase detector for SRTS values is given in Figure 8. It consists of delay registers 21, 22 comparators 23, 24, up/down counter 20, difference circuit 27, accumulator 25, and output register 26.

Because the "carry" of the RTS is not available from the counter 20, it has to be regenerated locally. This can be done by comparing the current RTS with the previous RTS stored in registers 21, 22. If the current RTS is smaller, a carry is generated. Bit errors in the received RTS may cause false carries and thereby cause cycle slips. A cycle slip results in wander on the output of  $3008/f_s$  seconds (for example, 1.47 ms @ E1 rates), which is unacceptable. An extra bit from the SRTS FIFO indicates whether an error was detected on the SRTS reassembly. In that case, both the corrupted received RTS and the local FITS is ignored by disabling the counter and the register.

This architecture has the advantage of offering phase recovery as well as frequency recovery; large input phase domain of  $2^n \text{UI}_{pp}$ , where n is the width of the counter in bits, large input jitter acceptance, and fast locking of the PLL with no creeping behaviour. Although it works directly on the residual time stamp, the circuitry in effect produces a phase output dependent on the difference between the actual timestamps.

For clock inputs, the phase detector block diagram is shown in Figure 9. This consists of difference circuits 30, 31, up/down counter 20, accumulator 25 and register 26. The

register 26 stores the phase output. This is clock by a signal  $f_{fb} \cdot n$ , where  $f_{fb}$  is the feedback signal an  $n$  is a multiplier, typically 8.  $n$  determines the granularity of the output.

The phase value is in the time domain now. Because the input clock has a relatively low frequency (compared to  $mclk$ ), a higher clock can be used to detect the phase difference between the input clock and the feedback clock resulting in decreased granularity and thus greater accuracy. This higher clock is also a feedback clock from the PLL 1. It samples the input clocks (synchronous detection) and therefore must be at least 2 times the highest input frequency. Because of jitter and frequency offsets on the input clock, this higher clock frequency is four times the nominal input clock frequency for J2, C4M and C8M and 8 times the nominal input clock frequency for DS1 and E1. This configuration has the same advantages as mentioned for the SRTS phase detector. The up/down counter 20 determines the jitter acceptance of the PLL 1 measured in UI (Unit Intervals). A jitter acceptance of 32 UI would mean a 5-bit counter. The counter does not wrap around to avoid the sawtooth effect noted above. This will increase the locking speed during phase acquisition. Of course the range of the accumulator must be such that no overflow occurs under any circumstances. For clock inputs, the same advantages of this architecture apply as for SRTS inputs.

The diagrams for the Phase Detector can be combined to one, as is shown in Figure 10. Up/down counter is passed through shifter 31 to register 26. The shifter adjusts the weight of the counter 20 to sixteen in SRTS mode and one in synchronous mode.

In the adaptive mode, the phase detector 10 is not used. A Circular Buffer in the reassembly engine 3 generates a signed 2's complement phaseword where phaseword = write\_pointer - read\_pointer - "average". "Average" is programmed by the user. This phaseword is used as the filter phase input instead of the phase output from the phase detector.

The loop filter 11 determines mainly the jitter transfer of the PLL 1. In addition, the PLL type, the locking range, and the phase alignment speed are determined in the loop filter 11. If a holdover mode is needed, it is built into the loop filter. The general block diagram is given in Figure 11. It consists of a limiter 40, adder 41, register 42, and adder 43. This filter has first order low pass behaviour. Because of the accumulator, the PLL is of type 2. The path starts with a limiter which limits the input phase value to approx. 40

ppm of the center frequency value and thereby limiting the phase alignment to approximately  $5 \text{ ns} / 125 \text{ } \mu\text{s}$ . A flag which indicates whether the limiter is actually limiting can be used for lock indication. Then the limited phase is accumulated (adder & register) to a frequency offset from the center frequency. The accumulator may not wrap  
5 around and its range determines the locking range of the PLL.

In holdover mode the phase, i.e. when LOS is active, the output of register 42 is no longer accumulated, but the value in the register 42 is used as the frequency offset. The I(ntegral) and P(roportional) values are added with different weights ( $P \gg I$ ). The difference in weight determines the damping of the loop. Because P is much larger than  
10 I, short term phase variations (jitter) are mainly corrected through the P path while long term phase variations (frequency variations) are mainly corrected through the I path.

The DCO 12 is basically an accumulator as shown in Figure 12. It consists of an adder 50, an adder 51, and a register 52. To modulate it, a frequency offset is added to the center frequency value. The carry output has the desired frequency which is a function of  
15 the accumulated value and the system clock frequency. For a given clock frequency and accumulator width, the output frequency can only have discrete values. Therefore, the desired output frequency has a limited accuracy.

The remaining value in the accumulator at a carry, the `rest_term`, represents the exact phase error of the carry pulse with respect to an ideal signal. The error is maximally  $1/f_{\text{sys}}$   
20 and is the intrinsic jitter of the DCO. Increasing  $f_{\text{sys}}$  reduces the intrinsic jitter. The `rest_term` can be used to correct the phase of the carry pulse, thereby reducing the intrinsic jitter.  $f_{\text{sys}}$  is high enough ( $> 64 \text{ MHz}$ ) to meet the jitter specs of 0.1 UI for DS1, E1 and J2 without much effort. An overview for a DCO with a clock frequency of 66 MHz is given in the table below.

### DCO Freerun Clocks and Accuracies.

	Clock	$f_s$ [MHz]	$f_{dco}$ [MHz]	Accuracy[ppm]
	DS1	1.544	12.352	0.05
	E1	2.048	16.384	-0.13
5	C4M	4.096	16.384	-0.13
	J2	6.312	25.284	-0.03
	C8M	8.192	32.768	0.05

The jitter reduction circuit 12 is shown in Figure 13. This consists of a comparator 55 and a DQ flip-flop 56. The intrinsic jitter of the DCO output is normally 1/mclk. With the use of the negative edge of the master clock, the intrinsic jitter can be reduced to half of that. When the rest term is smaller than half the center frequency value, the carry pulse is delayed by half a master clock cycle. When the rest term is greater than or equals half the center frequency value the carry pulse is not delayed.

In the case that mclk = 66 MHz with a 40-60% duty cycle, the output jitter would be maximally 9 ns. The spectrum of the jitter ranges from 0 Hz to  $f_{dco}/2$ . With this jitter reduction circuit, the following intrinsic output jitter values would be achieved:

#### Intrinsic Output Jitter @ mclk = 66 MHz, 40-60% duty cycle.

	Clock	DCO Output Jitter [UI]	Reduced Output Jitter [UI]
	DS1	0.02	0.01
20	E1	0.03	0.02
	C4M	0.06	0.04
	C8M	0.12	0.07
	J2	0.09	0.06

The output divider 13 divides the DCO output frequency down to the required outputs. A standard counter is necessary to do the job. For the generation of the 8 kHz output, a loadable counter is needed to do the division from E1/DS1/J2 to 8 kHz. Some additional circuitry can be added to generate the right frame pulse format.

The described clock recovery circuit is efficient and versatile, capable of operating in four modes. It can be used with or without multiple selectable center frequencies, with or without a center frequency programmed as a register from an microprocessor, with or

without a mode where the integrating register can be read by a microprocessor, with or without a mode where a microprocessor can adjust the center frequency based on an algorithm tied to the values read from the integration register (i.e. a mode where a microprocessor is used as a controller of the feedback loop of the PLL, giving longer time

5 constants than can easily be achieved in hardware), with or without a jitter reduction circuit, with different master clock (MCLK) frequencies, and with a jitter reduction circuit based on the negative clock edge, or with a jitter reduction circuit based on a tapped delay line. Various PLL parameters can be changed without departing from the spirit of the present invention, including: center frequency, locking range, input jitter

10 tolerance, jitter transfer function, max phase slope. It can be used with or without automatic switch-over to holdover passed on validity of received SRTS nibbles, and with or without automatic switch-over to holdover passed on underun due to lost ATM connection.

It can also be used for mode conversion, as an intermediate node, to convert received

15 adaptive clocking to transmit SRTS clocking, using the circuit of Figure 5, by making the input the adaptive input. It can be used for T1 to E1 conversion with an extra circuit, and it is also applicable to DS3 and E3 rates. It can also work in conjunction with another PLL (internal or external to the IC).